Self Equivalence of the Alternating Direction Method of Multipliers

Ming Yan* Wotao Yin*

August 11, 2014

Abstract

In this paper, we show interesting self equivalence results for the alternating direction method of multipliers (ADM or ADMM). Specifically, we show that ADM on a primal problem is equivalent to ADM on its Lagrange dual problem; ADM is equivalent to a primal-dual algorithm applied to a saddle-point formulation of the problem; when one of the two objective functions is quadratic with an affine domain, we can swap the update order of the two variables in ADM and obtain an equivalent algorithm. An example in extended monotropic programming is given to demonstrate that the primal-dual algorithm may be preferable over the other equivalent algorithms for its lower per-iteration complexity and, in the setting of distributed computation, better load balancing.

Keywords: Alternating Direction Method of Multipliers (ADM/ADMM), Douglas-Rachford Splitting (DRS), Primal-Dual Algorithm, Extended Monotropic Programming, Total Variation

1 Introduction

The alternating direction method of multipliers (ADM or ADMM) is very effective at solving complicated convex optimization problems. It applies to linearly-constrained convex optimization problems with separable objective functions in the following form:

$$\begin{cases} \underset{\mathbf{x},\mathbf{y}}{\text{minimize}} & f(\mathbf{x})+g(\mathbf{y}) \\ \text{subject to} & \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}=\mathbf{b}, \end{cases} \tag{P1}$$

where f, g are proper, closed, convex functions (may not be differentiable) and $\mathbf{A}, \mathbf{B}$ are linear mappings. ADM has been applied to both the primal and dual problems in many applications. For example, it was applied to both the primal and dual problems for ℓ_1 minimization in [20, 19]. As another example, ADM was applied to the dual problem in [12] and to the corresponding primal problem in [11].

In this paper, we show the following equivalence results for ADM:

1. It is equivalent to apply ADM to either the original form or the Lagrange dual of (P1).
2. ADM on either (P1) or its dual is equivalent to a primal-dual algorithm applied to a saddle-point formulation of (P1); in the latter algorithm, since one of the primal variables is hidden, each iteration may have a lower complexity than the other equivalent ones, as we shall demonstrate by an example.
3. Whenever either f or g is a quadratic (or, affine or linear) function, defined on either the whole space or an affine domain, swapping the order of $\mathbf{x}$ and $\mathbf{y}$ in ADM yields an equivalent algorithm.

*Department of Mathematics, University of California, Los Angeles, CA 90095, USA. Emails: yanm@math.ucla.edu and wotaoyin@math.ucla.edu

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
11 AUG 2014		**00-00-2014 to 00-00-2014**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Self Equivalence of the Alternating Direction Method of Multipliers	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California, Los Angeles,Department of Mathematics,Los Angeles,CA,90095**	8. PERFORMING ORGANIZATION REPORT NUMBER **CAM14-59**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In this paper, we show interesting self equivalence results for the alternating direction method of multipliers (ADM or ADMM). Speci cally, we show that ADM on a primal problem is equivalent to ADM on its Lagrange dual problem; ADM is equivalent to a primal-dual algorithm applied to a saddle- point formulation of the problem; when one of the two objective functions is quadratic with an a ne domain, we can swap the update order of the two variables in ADM and obtain an equivalent algorithm. An example in extended monotropic programming is given to demonstrate that the primal-dual algorithm may be preferable over the other equivalent algorithms for its lower per-iteration complexity and, in the setting of distributed computation, better load balancing.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**17**	

In all the three cases, given the iterates of one algorithm, we can recover the iterates of the equivalent algorithm by properly setting the initial iterates of the latter.

1.1 Notation and assumptions

Let $\mathcal{H}_1$, $\mathcal{H}_2$, and $\mathcal{G}$ be (possibly infinite dimensional) Hilbert spaces. Bold lowercase letters such as $\mathbf{x}$, $\mathbf{y}$, $\mathbf{u}$, and $\mathbf{v}$ are used for points in the Hilbert spaces. In the example of (P1), we have $\mathbf{x} \in \mathcal{H}_1$, $\mathbf{y} \in \mathcal{H}_2$, and $\mathbf{b} \in \mathcal{G}$. When the Hilbert space a point belongs to is clear from the context, we do not specify it for the sake of simplicity. The inner product between points $\mathbf{x}$ and $\mathbf{y}$ is denoted by $\langle \mathbf{x}, \mathbf{y} \rangle$, and $\|\mathbf{x}\|_2 := \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$ is the corresponding norm. $\|\cdot\|_1$ and $\|\cdot\|_\infty$ denote the ℓ_1 and ℓ_∞ norms, respectively. Bold uppercase letters such as $\mathbf{A}$ and $\mathbf{B}$ are used for both continuous linear mappings and matrices. $\mathbf{A}^*$ denotes the adjoint of $\mathbf{A}$. $\mathbf{I}$ denotes the identity map.

Both lower and upper case letters such as f, g, F, and G are used for functions. ι_C denotes the indicator function of the set C, which is assumed to be convex and nonempty. ι_C is defined as follows:

$$\iota_C(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x} \in C, \\ \infty, & \text{if } \mathbf{x} \notin C. \end{cases}$$

We make the following assumption throughout the paper:

Assumption 1. *Functions in this paper are assumed to be proper, closed, and convex. The saddle-point solutions to all the optimization problems in this paper are assumed to exist.*

Let $\partial f(\mathbf{x})$ be the subdifferential of function f at $\mathbf{x}$. The proximal operator $\mathbf{prox}_{f(\cdot)}$ is defined as

$$\mathbf{prox}_{f(\cdot)}(\mathbf{x}) = \arg\min_{\mathbf{y}} f(\mathbf{y}) + \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2,$$

where the minimization has the unique solution. The convex conjugate f^* of function f is defined as

$$f^*(\mathbf{v}) = \sup_{\mathbf{x}} \{\langle \mathbf{v}, \mathbf{x} \rangle - f(\mathbf{x})\}.$$

Let $\mathcal{P}_{B_1^\infty}$ be the projection onto the unit ℓ_∞ "ball" $B_1^\infty := \{\mathbf{x} : \|\mathbf{x}\|_\infty \leq 1\}$.

1.2 Organizations

This paper is organized as follows. The three equivalence results for ADM are shown in sections 2, 3,and 4: The primal-dual equivalence is discussed in sections 2; ADM is shown to be equivalent to a primal-dual algorithm applied to the saddle-point formulation in section 3; In section 4, we show that swapping the order of $\mathbf{x}$ and $\mathbf{y}$ in ADM yields an equivalent algorithm if f or g satisfies the condition mentioned above. We conclude this paper with two applications of our results: extended monotropic programming in section 5 and total variation image denoising in section 6.

2 Equivalence of ADM on primal and dual problems

In this section we show that ADM applied to (P1) is equivalent to it applied to the Lagrange dual of (P1).

Algorithm 1 describes how ADM is applied to (P1) [13, 14].

Algorithm 1 ADM on (P1)

initialize $\mathbf{x}_1^0$, $\mathbf{z}_1^0$, $\lambda > 0$

for $k = 0, 1, \cdots$ **do**

$\mathbf{y}_1^{k+1} = \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x}_1^k + \mathbf{B}\mathbf{y} - \mathbf{b} + \lambda\mathbf{z}_1^k\|_2^2$

$\mathbf{x}_1^{k+1} = \arg\min_{\mathbf{x}} f(\mathbf{x}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b} + \lambda\mathbf{z}_1^k\|_2^2$

$\mathbf{z}_1^{k+1} = \mathbf{z}_1^k + \lambda^{-1}(\mathbf{A}\mathbf{x}_1^{k+1} + \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b})$

end for

A *primal formulation* equivalent to (P1) is

$$\begin{cases} \underset{\mathbf{s},\mathbf{t}}{\text{minimize}} & F(\mathbf{s}) + G(\mathbf{t}) \\ \text{subject to} & \mathbf{s} + \mathbf{t} = \mathbf{0}, \end{cases} \tag{P2}$$

where $\mathbf{s}, \mathbf{t} \in \mathcal{G}$ and

$$F(\mathbf{s}) := \min_{\mathbf{x}} f(\mathbf{x}) + \iota_{\{\mathbf{x}:\mathbf{A}\mathbf{x}=\mathbf{s}\}}(\mathbf{x}), \tag{1a}$$

$$G(\mathbf{t}) := \min_{\mathbf{y}} g(\mathbf{y}) + \iota_{\{\mathbf{y}:\mathbf{B}\mathbf{y}-\mathbf{b}=\mathbf{t}\}}(\mathbf{y}). \tag{1b}$$

Remark 1. *If we define L_f and L_g as $L_f(\mathbf{x}) = \mathbf{A}\mathbf{x}$ and $L_g(\mathbf{y}) = \mathbf{B}\mathbf{y} - \mathbf{b}$, respectively, then F and G are known as the* infimal postcompositions *of f and g by L_f and L_f, respectively, according to [1, Def. 12.33]. They are written as*

$$F = L_f \triangleright f, \qquad G = L_g \triangleright g.$$

Algorithm 2 gives ADM applied to (P2). We will show Algorithms 1 and 2 are (trivially) equivalent.

Algorithm 2 ADM on (P2)

initialize $\mathbf{s}_2^0$, $\mathbf{z}_2^0$, $\lambda > 0$

for $k = 0, 1, \cdots$ **do**

$\mathbf{t}_2^{k+1} = \arg\min_{\mathbf{t}} G(\mathbf{t}) + (2\lambda)^{-1}\|\mathbf{s}_2^k + \mathbf{t} + \lambda\mathbf{z}_2^k\|_2^2$

$\mathbf{s}_2^{k+1} = \arg\min_{\mathbf{s}} F(\mathbf{s}) + (2\lambda)^{-1}\|\mathbf{s} + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k\|_2^2$

$\mathbf{z}_2^{k+1} = \mathbf{z}_2^k + \lambda^{-1}(\mathbf{s}_2^{k+1} + \mathbf{t}_2^{k+1})$

end for

The Lagrange dual problem to (P1) is

$$\underset{\mathbf{v}}{\text{minimize}} \quad f^*(-\mathbf{A}^*\mathbf{v}) + g^*(-\mathbf{B}^*\mathbf{v}) + \langle \mathbf{v}, \mathbf{b}\rangle, \tag{2}$$

which can derived from $\min_{\mathbf{v}} (-\min_{\mathbf{x},\mathbf{y}} L(\mathbf{x}, \mathbf{y}, \mathbf{v}))$ on the Lagrangian:

$$L(\mathbf{x}, \mathbf{y}, \mathbf{v}) = f(\mathbf{x}) + g(\mathbf{y}) + \langle \mathbf{v}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} - \mathbf{b}\rangle.$$

An *ADM-ready* formulation of (2) is

$$\begin{cases} \underset{\mathbf{u},\mathbf{v}}{\text{minimize}} & f^*(-\mathbf{A}^*\mathbf{u}) + g^*(-\mathbf{B}^*\mathbf{v}) + \langle \mathbf{v}, \mathbf{b}\rangle \\ \text{subject to} & \mathbf{u} - \mathbf{v} = \mathbf{0}. \end{cases} \tag{D1}$$

When ADM is applied to an ADM-ready formulation of a Lagrange dual problem, we call it *Dual ADM*. The original ADM is called *Primal ADM*.

Following similar steps, the ADM ready formulation of the Lagrange dual to (P2) is

$$\begin{cases} \underset{\mathbf{u},\mathbf{v}}{\text{minimize}} & F^*(-\mathbf{u}) + G^*(-\mathbf{v}) \\ \text{subject to} & \mathbf{u} - \mathbf{v} = \mathbf{0}. \end{cases} \tag{D2}$$

The equivalence between (D1) and (D2) is trivial since

$$\begin{aligned} F^*(\mathbf{u}) &= f^*(\mathbf{A}^*\mathbf{u}) \\ G^*(\mathbf{v}) &= g^*(\mathbf{B}^*\mathbf{v}) - \langle \mathbf{v}, \mathbf{b} \rangle, \end{aligned}$$

which follows from Lemma 1.

Lemma 1. *If L is affine and can be expressed as $L(\cdot) = \mathbf{A} \cdot + \mathbf{b}$, the convex conjugate of $L \triangleright f$, the infimal postcomposition of f by L, can be found as follows:*

$$(L \triangleright f)^*(\cdot) = f^*(\mathbf{A}^* \cdot) + \langle \cdot, \mathbf{b} \rangle.$$

Proof. Following from the definitions of convex conjugate and infimal postcomposition, we have

$$\begin{aligned} (L \triangleright f)^*(\mathbf{v}) &= \sup_{\mathbf{y}} \langle \mathbf{v}, \mathbf{y} \rangle - L \triangleright f(\mathbf{y}) = \sup_{\mathbf{x}} \langle \mathbf{v}, \mathbf{A}\mathbf{x} + \mathbf{b} \rangle - f(\mathbf{x}) \\ &= \sup_{\mathbf{x}} \langle \mathbf{A}^*\mathbf{v}, \mathbf{x} \rangle - f(\mathbf{x}) + \langle \mathbf{v}, \mathbf{b} \rangle = f^*(\mathbf{A}^*\mathbf{v}) + \langle \mathbf{v}, \mathbf{b} \rangle. \end{aligned}$$

□

We apply ADM on (D1)/(D2) in Algorithm 3.

Algorithm 3 ADM on (D1)/(D2)

initialize $\mathbf{u}_3^0$, $\mathbf{z}_3^0$, $\lambda > 0$
for $k = 0, 1, \cdots$ **do**
$\mathbf{v}_3^{k+1} = \arg\min_{\mathbf{v}} G^*(-\mathbf{v}) + \frac{\lambda}{2} \|\mathbf{u}_3^k - \mathbf{v} + \lambda^{-1}\mathbf{z}_3^k\|_2^2$
$\mathbf{u}_3^{k+1} = \arg\min_{\mathbf{u}} F^*(-\mathbf{u}) + \frac{\lambda}{2} \|\mathbf{u} - \mathbf{v}_3^{\mathbf{k+1}} + \lambda^{-1}\mathbf{z}_3^k\|_2^2$
$\mathbf{z}_3^{k+1} = \mathbf{z}_3^k + \lambda(\mathbf{u}_3^{k+1} - \mathbf{v}_3^{k+1})$
end for

The following equivalence is shown in Theorem 1.

$$\boxed{\text{ADM on (P1)}} \Longleftrightarrow \boxed{\text{ADM on (P2)}} \Longleftrightarrow \boxed{\text{ADM on (D1)/(D2)}}$$

Theorem 1 (Equivalence of Algorithms 1-3)**.** *Suppose $\mathbf{A}\mathbf{x}_1^0 = \mathbf{s}_2^0 = \mathbf{z}_3^0$ and $\mathbf{z}_1^0 = \mathbf{z}_2^0 = \mathbf{u}_3^0$ and that the same parameter λ is used in Algorithms 1-3. Then, their equivalence can be established as follows:*

1. *From $\mathbf{x}_1^k$, $\mathbf{y}_1^k$, $\mathbf{z}_1^k$ of Algorithm 1, we obtain $\mathbf{t}_2^k$, $\mathbf{s}_2^k$, $\mathbf{z}_2^k$ of Algorithm 2 through:*

$$\mathbf{t}_2^k = \mathbf{B}\mathbf{y}_1^k - \mathbf{b}, \tag{3a}$$
$$\mathbf{s}_2^k = \mathbf{A}\mathbf{x}_1^k, \tag{3b}$$
$$\mathbf{z}_2^k = \mathbf{z}_1^k. \tag{3c}$$

From $\mathbf{t}_2^k$, $\mathbf{s}_2^k$, $\mathbf{z}_2^k$ of Algorithm 2, we obtain $\mathbf{y}_1^k$, $\mathbf{x}_1^k$, $\mathbf{z}_1^k$ of Algorithm 1 through:

$$\mathbf{y}_1^k = \arg\min_{\mathbf{y}} \{g(\mathbf{y}) : \mathbf{B}\mathbf{y} - \mathbf{b} = \mathbf{t}_2^k\}, \tag{4a}$$

$$\mathbf{x}_1^k = \arg\min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{A}\mathbf{x} = \mathbf{s}_2^k\}, \tag{4b}$$

$$\mathbf{z}_1^k = \mathbf{z}_2^k. \tag{4c}$$

2. We can recover the iterates of Algorithm 2 and 3 from each other through

$$\mathbf{u}_3^k = \mathbf{z}_2^k, \qquad \mathbf{z}_3^k = \mathbf{s}_2^k. \tag{5}$$

Proof. Part 1. Proof by induction.
We argue that under (3b) and (3c), Algorithms 1 and 2 have *essentially identical* subproblems in their *first* steps at the kth iteration. Consider the following problem, which is obtained by plugging the definition of $G(\cdot)$ into the $\mathbf{t}_2^{k+1}$-subproblem of Algorithm 2:

$$(\mathbf{y}_1^{k+1}, \mathbf{t}_2^{k+1}) = \arg\min_{\mathbf{y},\mathbf{t}} g(\mathbf{y}) + \iota_{\{(\mathbf{y},\mathbf{t}):\mathbf{B}\mathbf{y}-\mathbf{b}=\mathbf{t}\}}(\mathbf{y},\mathbf{t}) + (2\lambda)^{-1}\|\mathbf{s}_2^k + \mathbf{t} + \lambda\mathbf{z}_2^k\|_2^2. \tag{6}$$

If one minimizes over $\mathbf{y}$ first while keeping $\mathbf{t}$ as a variable, one eliminates $\mathbf{y}$ and recovers the $\mathbf{t}_2^{k+1}$-subproblem of Algorithm 2. If one minimizes over $\mathbf{t}$ first while keeping $\mathbf{y}$ as a variable, then after plugging in (3b) and (3c), problem (6) reduces to the $\mathbf{y}_1^{k+1}$-subproblem of Algorithm 1. In addition, $(\mathbf{y}_1^{k+1}, \mathbf{t}_2^{k+1})$ obeys

$$\mathbf{t}_2^{k+1} = \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b}, \tag{7}$$

which is (3a) at $k+1$. Plugging $\mathbf{t} = \mathbf{t}_2^{k+1}$ into (6) yields problem (4a) for $\mathbf{y}_1^{k+1}$, which must be equivalent to the $\mathbf{y}_1^{k+1}$-subproblem of Algorithm 2. Therefore, the $\mathbf{y}_1^{k+1}$-subproblem of Algorithm 1 and the $\mathbf{t}_2^{k+1}$-subproblem of Algorithm 2 are equivalent through (3a) and (4a) at $k+1$, respectively.

Similarly, under (7) and (3c), we can show that the $\mathbf{x}_1^{k+1}$-subproblem of Algorithm 1 and the $\mathbf{s}_2^{k+1}$-subproblem of Algorithm 2 are equivalent through the formulas for (3b) and (4b) at $k+1$, respectively.

Finally, under (3a) and (3b) at $k+1$ and $\mathbf{z}_2^k = \mathbf{z}_1^k$, the formulas for $\mathbf{z}_1^{k+1}$ and $\mathbf{z}_2^{k+1}$ in Algorithms 1 and 2 are identical, and they return $\mathbf{z}_1^{k+1} = \mathbf{z}_2^{k+1}$, which is (3c) and (4c) at $k+1$.

Part 2. Proof by induction. Suppose that (5) holds. We shall show that (5) holds at $k+1$. Starting from the optimality condition of the $\mathbf{t}_2^{k+1}$-subproblem of Algorithm 2, we derive

$$\begin{aligned}
&\mathbf{0} \in \partial G(\mathbf{t}_2^{k+1}) + \lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k) \\
\iff &\mathbf{t}_2^{k+1} \in \partial G^*(-\lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k)) \\
\iff &\lambda\left[\lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k)\right] - (\lambda\mathbf{z}_2^k + \mathbf{s}_2^k) \in \partial G^*(-\lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k)) \\
\iff &-\lambda\left[\lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k)\right] + (\lambda\mathbf{u}_3^k + \mathbf{z}_3^k) \in -\partial G^*(-\lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k)) \\
\iff &\mathbf{0} \in -\partial G^*(-\lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k)) - \lambda\left[\mathbf{u}_3^k - \lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k) + \lambda^{-1}\mathbf{z}_3^k\right] \\
\iff &\mathbf{v}_3^{k+1} = \lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k) = \lambda^{-1}(\mathbf{z}_3^k + \mathbf{t}_2^{k+1} + \lambda\mathbf{z}_2^k),
\end{aligned}$$

where the last equivalence follows from the optimality condition for the $\mathbf{v}_3^{k+1}$-subproblem of Algorithm 3.

Starting from the optimality condition of the $\mathbf{s}_2^{k+1}$-subproblem of Algorithm 2, and applying the update,

$\mathbf{z}_2^{k+1} = \mathbf{z}_2^k + \lambda^{-1}(\mathbf{s}_2^{k+1} + \mathbf{t}_2^{k+1})$, in Algorithm 2 and the identity of $\mathbf{t}_2^{k+1}$ obtained above, we derive

$$\begin{aligned}
&\mathbf{0} \in \partial F(\mathbf{s}_2^{k+1}) + \lambda^{-1}(\mathbf{s}_2^{k+1} + \mathbf{t}_2^{k+1} + \lambda \mathbf{z}_2^k) \\
\Longleftrightarrow\ &\mathbf{0} \in \partial F(\mathbf{s}_2^{k+1}) + \mathbf{z}_2^{k+1} \\
\Longleftrightarrow\ &\mathbf{0} \in \mathbf{s}_2^{k+1} - \partial F^*(-\mathbf{z}_2^{k+1}) \\
\Longleftrightarrow\ &\mathbf{0} \in \lambda(\mathbf{z}_2^{k+1} - \mathbf{z}_2^k) - \mathbf{t}_2^{k+1} - \partial F^*(-\mathbf{z}_2^{k+1}) \\
\Longleftrightarrow\ &\mathbf{0} \in \lambda(\mathbf{z}_2^{k+1} - \mathbf{z}_2^k) + \mathbf{z}_3^k + \lambda(\mathbf{z}_2^k - \mathbf{v}_3^{k+1}) - \partial F^*(-\mathbf{z}_2^{k+1}) \\
\Longleftrightarrow\ &\mathbf{0} \in -\partial F^*(-\mathbf{z}_2^{k+1}) + \lambda(\mathbf{z}_2^{k+1} - \mathbf{v}_3^{k+1} + \lambda^{-1}\mathbf{z}_3^k) \\
\Longleftrightarrow\ &\mathbf{z}_2^{k+1} = \mathbf{u}_3^{k+1}.
\end{aligned}$$

where the last equivalence follows from the optimality condition for the $\mathbf{u}_3^{k+1}$-subproblem of Algorithm 3. Finally, combining the update formulas of $\mathbf{z}_2^{k+1}$ and $\mathbf{z}_3^{k+1}$ in Algorithm 2 and 3, respectively, as well as the identities for $\mathbf{u}_3^{k+1}$ and $\mathbf{v}_3^{k+1}$ obtained above, we obtain

$$\begin{aligned}
\mathbf{z}_3^{k+1} &= \mathbf{z}_3^k + \lambda(\mathbf{u}_3^{k+1} - \mathbf{v}_3^{k+1}) = \mathbf{s}^k + \lambda(\mathbf{z}_2^{k+1} - \mathbf{z}_2^k - \lambda^{-1}(\mathbf{s}_2^k + \mathbf{t}_2^{k+1})) \\
&= \lambda(\mathbf{z}_2^{k+1} - \mathbf{z}_2^k) - \mathbf{t}_2^{k+1} = \mathbf{s}_2^{k+1}.
\end{aligned}$$

□

Remark 2. *Following Part 1 of the theorem, we can view problem* (P2) *as the* master problem *of* (P1)*, whereas the two subproblems in* (1) *are independent. We can say that ADM is essentially an algorithm applied only to the master problem* (P2)*, which is Algorithm 2; this fact has been obscured by the often-seen Algorithm 1, which integrates ADM on the master problem with the independent subproblems.*

Part 2 of the theorem shows that ADM is a symmetric primal-dual algorithm. The reciprocal positions of parameter λ indicates its function to "balance" the primal and dual progresses.

Remark 3. *ADM's primal-dual equivalence can also be derived by combining the following two equivalence results: (i) the equivalence between ADM on the primal problem and the Douglas-Rachford splitting (DRS) algorithm [7, 16] on the dual problem [12], and (ii) the equivalence result between DRS algorithms applied to the master problem* (P2) *and its dual problem (cf. [8, Chapter 3.5][9]). In this paper, however, we provide an elementary algebraic proof in order to derive the formulas in theorem 1 that recover the iterates of one algorithm from another.*

Next we give two concrete examples that illustrate the equivalence.

2.1 Example: basis pursuit

The basis pursuit problem seeks for the minimal ℓ_1 solution to a set of linear equations:

$$\underset{\mathbf{u}}{\text{minimize}}\, \|\mathbf{u}\|_1 \quad \text{subject to } \mathbf{A}\mathbf{u} = \mathbf{b}. \tag{8}$$

Its Lagrange dual problem is

$$\underset{\mathbf{x}}{\text{minimize}}\, -\mathbf{b}^T\mathbf{x} \quad \text{subject to } \|\mathbf{A}^*\mathbf{x}\|_\infty \le 1. \tag{9}$$

The YALL1 algorithms [20] implement ADMs on a set of primal and dual formulations for basis pursuit and LASSO, yet ADM for (8) is not given (however, a linearized ADM is given for (8)). Although seemingly awkward, problem (8) can be turned equivalently into the ADM-ready form

$$\underset{\mathbf{u},\mathbf{v}}{\text{minimize}}\, \|\mathbf{v}\|_1 + \iota_{\{\mathbf{u}:\mathbf{A}\mathbf{u}=\mathbf{b}\}}(\mathbf{u}) \quad \text{subject to } \mathbf{u} - \mathbf{v} = \mathbf{0}. \tag{10}$$

Similarly, problem (9) can be turned equivalently into the ADM-ready form

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} -\mathbf{b}^T\mathbf{x} + \iota_{B_1^\infty}(\mathbf{y}) \quad \text{subject to } \mathbf{A}^*\mathbf{x} - \mathbf{y} = \mathbf{0}. \tag{11}$$

For simplicity, let us suppose that $\mathbf{A}$ has full row rank so the inverse of $\mathbf{A}\mathbf{A}^*$ exists. (Otherwise, $\mathbf{A}\mathbf{u} = \mathbf{b}$ are redundant whenever they are consistent; and $(\mathbf{A}\mathbf{A}^*)^{-1}$ shall be replaced by the pseudo-inverse below.) ADM for problem (10) can be simplified to the iteration:

$$\mathbf{v}_3^{k+1} = \arg\min_{\mathbf{v}} \|\mathbf{v}\|_1 + \frac{\lambda}{2}\|\mathbf{u}_3^k - \mathbf{v} + \frac{1}{\lambda}\mathbf{z}_3^k\|_2^2, \tag{12a}$$

$$\mathbf{u}_3^{k+1} = \mathbf{v}_3^{k+1} - \frac{1}{\lambda}\mathbf{z}_3^k - \mathbf{A}^*(\mathbf{A}\mathbf{A}^*)^{-1}(\mathbf{A}(\mathbf{v}_3^{k+1} - \frac{1}{\lambda}\mathbf{z}_3^k) - \mathbf{b}), \tag{12b}$$

$$\mathbf{z}_3^{k+1} = \mathbf{z}_3^k + \lambda(\mathbf{u}_3^{k+1} - \mathbf{v}_3^{k+1}). \tag{12c}$$

ADM for problem (11) can be simplified to the iteration:

$$\mathbf{y}_1^{k+1} = \mathcal{P}_{B_1^\infty}(\mathbf{A}^*\mathbf{x}_1^k + \lambda\mathbf{z}_1^k), \tag{13a}$$

$$\mathbf{x}_1^{k+1} = (\mathbf{A}\mathbf{A}^*)^{-1}(\mathbf{A}\mathbf{y}_1^{k+1} - \lambda(\mathbf{A}\mathbf{z}_1^k - \mathbf{b})), \tag{13b}$$

$$\mathbf{z}_1^{k+1} = \mathbf{z}_1^k + \lambda^{-1}(\mathbf{A}^*\mathbf{x}_1^{k+1} - \mathbf{y}_1^{k+1}). \tag{13c}$$

The corollary below follows directly from Theorem 1 by associating (11) and (10) as (P1) and (D2), and (13) and (12) with the iterations of Algorithms 1 and 3, respectively.

Corollary 1. *Suppose that* $\mathbf{A}\mathbf{u} = \mathbf{b}$ *are consistent. Consider ADM iterations* (12) *and* (13)*. Let* $\mathbf{u}_3^0 = \mathbf{z}_1^0$ *and* $\mathbf{z}_3^0 = \mathbf{A}^*\mathbf{x}_1^0$*. Then, for* $k \geq 1$*, iterations* (12) *and* (13) *are equivalent. In particular,*

- *From* $\mathbf{x}_1^k$, $\mathbf{z}_1^k$ *in* (13)*, we obtain* $\mathbf{u}_3^k$, $\mathbf{z}_3^k$ *in* (12) *through:*

$$\mathbf{u}_3^k = \mathbf{z}_1^k, \qquad \mathbf{z}_3^k = \mathbf{A}^*\mathbf{x}_1^k.$$

- *From* $\mathbf{u}_3^k$, $\mathbf{z}_3^k$ *in* (12)*, we obtain* $\mathbf{x}_1^k$, $\mathbf{z}_1^k$ *in* (13) *through:*

$$\mathbf{x}_1^k = (\mathbf{A}\mathbf{A}^*)^{-1}\mathbf{A}\mathbf{z}_3^k, \qquad \mathbf{z}_1^k = \mathbf{u}_3^k.$$

2.2 Example: basis pursuit denoising

The basis pursuit denoising problem is

$$\underset{\mathbf{u}}{\text{minimize}} \|\mathbf{u}\|_1 + \frac{1}{2\alpha}\|\mathbf{A}\mathbf{u} - \mathbf{b}\|_2^2 \tag{14}$$

and its Lagrange dual problem, in the ADM-ready form, is

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} -\langle \mathbf{b}, \mathbf{x}\rangle + \frac{\alpha}{2}\|\mathbf{x}\|_2^2 + \iota_{B_1^\infty}(\mathbf{y}) \quad \text{subject to } \mathbf{A}^*\mathbf{x} - \mathbf{y} = \mathbf{0}. \tag{15}$$

The iteration of ADM for (15) is

$$\mathbf{y}_1^{k+1} = \mathcal{P}_{B_1^\infty}(\mathbf{A}^*\mathbf{x}_1^k + \lambda\mathbf{z}_1^k), \tag{16a}$$

$$\mathbf{x}_1^{k+1} = (\mathbf{A}\mathbf{A}^* + \alpha\lambda\mathbf{I})^{-1}(\mathbf{A}\mathbf{y}_1^{k+1} - \lambda(\mathbf{A}\mathbf{z}_1^k - \mathbf{b})), \tag{16b}$$

$$\mathbf{z}_1^{k+1} = \mathbf{z}_1^k + \lambda^{-1}(\mathbf{A}^*\mathbf{x}_1^{k+1} - \mathbf{y}_1^{k+1}). \tag{16c}$$

The ADM-ready form of the original problem (14) is

$$\underset{\mathbf{u},\mathbf{v}}{\text{minimize}}\, \|\mathbf{v}\|_1 + \frac{1}{2\alpha}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 \quad \text{subject to } \mathbf{u}-\mathbf{v}=\mathbf{0}, \tag{17}$$

whose ADM iteration is

$$\mathbf{v}_3^{k+1} = \arg\min_{\mathbf{v}} \|\mathbf{v}\|_1 + \frac{\lambda}{2}\|\mathbf{u}_3^k - \mathbf{v} + \frac{1}{\lambda}\mathbf{z}_3^k\|_2^2 \tag{18a}$$

$$\mathbf{u}_3^{k+1} = (\mathbf{A}^*\mathbf{A} + \alpha\lambda\mathbf{I})^{-1}(\mathbf{A}^*\mathbf{b} + \alpha\lambda\mathbf{v}_3^{k+1} - \alpha\mathbf{z}_3^k) \tag{18b}$$

$$\mathbf{z}_3^{k+1} = \mathbf{z}_3^k + \lambda(\mathbf{u}_3^{k+1} - \mathbf{v}_3^{k+1}) \tag{18c}$$

The corollary below follows directly from Theorem 1.

Corollary 2. *Consider ADM iterations* (16) *and* (18). *Let* $\mathbf{u}_3^0 = \mathbf{z}_1^0$ *and* $\mathbf{z}_3^0 = \mathbf{A}^*\mathbf{x}_1^0$. *For* $k \geq 1$, *ADM on the dual and primal problems* (16) *and* (18) *are equivalent in the following way:*

- *From* $\mathbf{x}_1^k$, $\mathbf{z}_1^k$ *in* (16), *we recover* $\mathbf{u}_3^k$, $\mathbf{z}_3^k$ *in* (18) *through:*

$$\mathbf{u}_3^k = \mathbf{z}_1^k, \qquad \mathbf{z}_3^k = \mathbf{A}^*\mathbf{x}_1^k.$$

- *From* $\mathbf{u}_3^k$, $\mathbf{z}_3^k$ *in* (18), *we recover* $\mathbf{x}_1^k$, $\mathbf{z}_1^k$ *in* (16) *through:*

$$\mathbf{x}_1^k = -(\mathbf{A}\mathbf{u}_3^k - \mathbf{b})/\alpha, \qquad \mathbf{z}_1^k = \mathbf{u}_3^k.$$

Remark 4. *Iteration* (18) *is different from that of ADM for another ADM-ready form of* (14)

$$\underset{\mathbf{u},\mathbf{v}}{\text{minimize}}\, \|\mathbf{u}\|_1 + \frac{1}{2\alpha}\|\mathbf{v}\|_2^2 \quad \text{subject to } \mathbf{A}\mathbf{u}-\mathbf{v}=\mathbf{b}, \tag{19}$$

which is used in [20]. In general, there are different ADM-ready forms and their ADM algorithms yield different iterates. ADM on one ADM-ready form is equivalent to it on the corresponding dual ADM-ready form.

3 ADM as a primal-dual algorithm on a saddle-point problem

As shown in section 2, ADM on a pair of convex primal and dual problems are equivalent, and there is a connection between $\mathbf{z}_1^k$ in Algorithm 1 and dual variable $\mathbf{u}_3^k$ in Algorithm 3. This primal-dual equivalence naturally suggests that ADM is also equivalent to a primal-dual algorithm involving both primal and dual variables.

We derive problem (P1) into an equivalent primal-dual saddle-point problem (21) as follows:

$$\begin{aligned}
&\min_{\mathbf{y},\mathbf{x}} g(\mathbf{y}) + f(\mathbf{x}) + \iota_{\{(\mathbf{x},\mathbf{y}):\mathbf{A}\mathbf{x}=\mathbf{b}-\mathbf{B}\mathbf{y}\}}(\mathbf{x},\mathbf{y}) \\
=&\min_{\mathbf{y}} g(\mathbf{y}) + F(\mathbf{b}-\mathbf{B}\mathbf{y}) \\
=&\min_{\mathbf{y}}\max_{\mathbf{u}} g(\mathbf{y}) + \langle -\mathbf{u}, \mathbf{b}-\mathbf{B}\mathbf{y}\rangle - F^*(-\mathbf{u}) \qquad (20)\\
=&\min_{\mathbf{y}}\max_{\mathbf{u}} g(\mathbf{y}) + \langle \mathbf{u}, \mathbf{B}\mathbf{y}-\mathbf{b}\rangle - f^*(-\mathbf{A}^*\mathbf{u}). \qquad (21)
\end{aligned}$$

A primal-dual algorithm for solving (21) is described in Algorithm 4. Theorem 2 establishes the equivalence between Algorithms 1 and 4.

Algorithm 4 Primal-dual formulation of ADM on Problem (21)

initialize $\mathbf{u}_5^0$, $\mathbf{u}_5^{-1}$, $\mathbf{y}_5^0$, $\lambda > 0$
for $k = 0, 1, \cdots$ **do**
$\bar{\mathbf{u}}_5^k = 2\mathbf{u}_5^k - \mathbf{u}_5^{k-1}$
$\mathbf{y}_5^{k+1} = \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\mathbf{B}\mathbf{y} - \mathbf{B}\mathbf{y}_5^k + \lambda\bar{\mathbf{u}}_5^k\|_2^2$
$\mathbf{u}_5^{k+1} = \arg\min_{\mathbf{u}} f^*(-\mathbf{A}^*\mathbf{u}) - \langle \mathbf{u}, \mathbf{B}\mathbf{y}_5^{k+1} - \mathbf{b}\rangle + \lambda/2\|\mathbf{u} - \mathbf{u}_5^k\|_2^2$
end for

Remark 5. *Paper [4] proposed a primal-dual algorithm for* (20) *and obtained the connection between ADM and that primal-dual algorithm [10]: When* $\mathbf{B} = \mathbf{I}$*, ADM is equivalent to the primal-dual algorithm in [4]; When* $\mathbf{B} \neq \mathbf{I}$*, the primal-dual algorithm is a preconditioned ADM as an additional proximal term* $\delta/2\|\mathbf{y} - \mathbf{y}_5^k\|_2^2 - (2\lambda)^{-1}\|\mathbf{B}\mathbf{y} - \mathbf{B}\mathbf{y}_5^k\|_2^2$ *is added to the subproblem for* $\mathbf{y}_5^{k+1}$*. This is also a special case of inexact ADM in [6]. Our Algorithm 4 is a primal-dual algorithm that is equivalent to ADM in the general case.*

Theorem 2 (Equivalence between Algorithms 1 and 4)**.** *Suppose that* $\mathbf{A}\mathbf{x}_1^0 = \lambda(\mathbf{u}_5^0 - \mathbf{u}_5^{-1}) + \mathbf{b} - \mathbf{B}\mathbf{y}_5^0$ *and* $\mathbf{z}_1^0 = \mathbf{u}_5^0$*. Then, Algorithms 1 and 4 are equivalent with the identities:*

$$\mathbf{A}\mathbf{x}_1^k = \lambda(\mathbf{u}_5^k - \mathbf{u}_5^{k-1}) + \mathbf{b} - \mathbf{B}\mathbf{y}_5^k, \qquad \mathbf{z}_1^k = \mathbf{u}_5^k, \tag{22}$$

for all $k > 0$*.*

Proof. By assumption, (22) holds at iteration $k = 0$.
Proof by induction. Suppose that (22) holds at iteration $k \geq 0$. We shall establish (22) at iteration $k + 1$. From the first step of Algorithm 1, we have

$$\begin{aligned}\mathbf{y}_1^{k+1} &= \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x}_1^k + \mathbf{B}\mathbf{y} - \mathbf{b} + \lambda\mathbf{z}_1^k\|_2^2 \\ &= \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\lambda(\mathbf{u}_5^k - \mathbf{u}_5^{k-1}) + \mathbf{B}\mathbf{y} - \mathbf{B}\mathbf{y}_5^k + \lambda\mathbf{u}_5^k\|_2^2,\end{aligned}$$

which is the same as the first step in Algorithm 4. Thus we have $\mathbf{y}_1^{k+1} = \mathbf{y}_5^{k+1}$.

Combing the second and third steps of Algorithm 1, we have

$$\mathbf{0} \in \partial f(\mathbf{x}_1^{k+1}) + \lambda^{-1}\mathbf{A}^*(\mathbf{A}\mathbf{x}_1^{k+1} + \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b} + \lambda\mathbf{z}_1^k) = \partial f(\mathbf{x}_1^{k+1}) + \mathbf{A}^*\mathbf{z}_1^{k+1}.$$

Therefore,

$$\begin{aligned}&\mathbf{x}_1^{k+1} \in \partial f^*(-\mathbf{A}^*\mathbf{z}_1^{k+1}) \\ \Longrightarrow\ &\mathbf{A}\mathbf{x}_1^{k+1} \in \partial F^*(-\mathbf{z}_1^{k+1}) \\ \Longleftrightarrow\ &\lambda(\mathbf{z}_1^{k+1} - \mathbf{z}_1^k) + \mathbf{b} - \mathbf{B}\mathbf{y}_1^{k+1} \in \partial F^*(-\mathbf{z}_1^{k+1}) \\ \Longleftrightarrow\ &\mathbf{z}_1^{k+1} = \arg\min_{\mathbf{z}} F^*(-\mathbf{z}) - \langle \mathbf{z}, \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b}\rangle + \lambda/2\|\mathbf{z} - \mathbf{z}_1^k\|_2^2 \\ \Longleftrightarrow\ &\mathbf{z}_1^{k+1} = \arg\min_{\mathbf{z}} f^*(-\mathbf{A}^*\mathbf{z}) - \langle \mathbf{z}, \mathbf{B}\mathbf{y}_5^{k+1} - \mathbf{b}\rangle + \lambda/2\|\mathbf{z} - \mathbf{u}_5^k\|_2^2,\end{aligned}$$

where the last line is the second step of Algorithm 4. Therefore, we have $\mathbf{z}_1^{k+1} = \mathbf{u}_5^{k+1}$ and $\mathbf{A}\mathbf{x}_1^{k+1} = \lambda(\mathbf{z}_1^{k+1} - \mathbf{z}_1^k) + \mathbf{b} - \mathbf{B}\mathbf{y}_1^{k+1} = \lambda(\mathbf{u}_5^{k+1} - \mathbf{u}_5^k) + \mathbf{b} - \mathbf{B}\mathbf{y}_5^{k+1}$. □

4 Equivalence of ADM for different orders

In both problem (P1) and Algorithm 1, we can swap $\mathbf{x}$ and $\mathbf{y}$ and obtain Algorithm 5 below, which is still an algorithm of ADM. In general, the two algorithms are different. In this section, we show that for a certain type of function f (or g), Algorithms 1 and 5 become equivalent.

Algorithm 5 ADM2 on (P1)

initialize $\mathbf{y}_4^0$, $\mathbf{z}_4^0$, $\lambda > 0$

for $k = 0, 1, \cdots$ **do**

$\mathbf{x}_4^{k+1} = \arg\min_{\mathbf{x}} f(\mathbf{x}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y}_4^k - \mathbf{b} + \lambda\mathbf{z}_4^k\|_2^2$

$\mathbf{y}_4^{k+1} = \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y} - \mathbf{b} + \lambda\mathbf{z}_4^k\|_2^2$

$\mathbf{z}_4^{k+1} = \mathbf{z}_4^k + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b})$

end for

The assumption that we need is that either $\mathbf{prox}_{F(\cdot)}$ or $\mathbf{prox}_{G(\cdot)}$ is affine (cf. (1) for the definitions of F and G).

Definition 1. *A mapping T is affine if, for any $\mathbf{r}_1$ and $\mathbf{r}_2$,*

$$T\left(\frac{1}{2}\mathbf{r}_1 + \frac{1}{2}\mathbf{r}_2\right) = \frac{1}{2}T\mathbf{r}_1 + \frac{1}{2}T\mathbf{r}_2.$$

Proposition 1. *Let $\lambda > 0$. The following statements are equivalent:*

1. $\mathbf{prox}_{G(\cdot)}$ *is affine;*
2. $\mathbf{prox}_{\lambda G(\cdot)}$ *is affine;*
3. $a\mathbf{prox}_{G(\cdot)} \circ b\mathbf{I} + c\mathbf{I}$ *is affine for any scalars a, b and c;*
4. $\mathbf{prox}_{G^*(\cdot)}$ *is affine;*
5. *G is convex quadratic (or, affine or constant) and its domain* $\mathrm{dom}(G)$ *is either $\mathcal{G}$ or the intersection of hyperplanes in $\mathcal{G}$.*

In addition, if function g is convex quadratic and its domain is the intersection of hyperplanes, then function G defined in (1b) *satisfies Part 5 above.*

Proposition 2. *If $\mathbf{prox}_{G(\cdot)}$ is affine, then the following holds for any $\mathbf{r}_1$ and $\mathbf{r}_2$:*

$$\mathbf{prox}_{G(\cdot)}(2\mathbf{r}_1 - \mathbf{r}_2) = 2\mathbf{prox}_{G(\cdot)}\mathbf{r}_1 - \mathbf{prox}_{G(\cdot)}\mathbf{r}_2. \tag{23}$$

Proof. Equation (23) is obtained by defining $\bar{\mathbf{r}}_1 = 2\mathbf{r}_1 - \mathbf{r}_2$ and $\bar{\mathbf{r}}_2 := \mathbf{r}_2$ and rearranging

$$\mathbf{prox}_{G(\cdot)}\left(\frac{1}{2}\bar{\mathbf{r}}_1 + \frac{1}{2}\bar{\mathbf{r}}_2\right) = \frac{1}{2}\mathbf{prox}_{G(\cdot)}\bar{\mathbf{r}}_1 + \frac{1}{2}\mathbf{prox}_{G(\cdot)}\bar{\mathbf{r}}_2.$$

□

Theorem 3 (Equivalence of Algorithms 1 and 5)**.**

1. *Assume that* $\mathbf{prox}_{\lambda G(\cdot)}$ *is affine. Given the sequences* $\mathbf{y}_4^k$, $\mathbf{z}_4^k$, *and* $\mathbf{x}_4^k$ *of Algorithm 5, if* $\mathbf{y}_4^0$ *and* $\mathbf{z}_4^0$ *satisfy* $-\mathbf{z}_4^0 \in \partial G(\mathbf{B}\mathbf{y}_4^0 - \mathbf{b})$, *then we can initialize Algorithm 1 with* $\mathbf{x}_1^0 = \mathbf{x}_4^1$ *and* $\mathbf{z}_1^0 = \mathbf{z}_4^0 + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^1 + \mathbf{B}\mathbf{y}_4^0 - \mathbf{b})$, *and recover the sequences* $\mathbf{x}_1^k$ *and* $\mathbf{z}_1^k$ *of Algorithm 1 through*

$$\mathbf{x}_1^k = \mathbf{x}_4^{k+1}, \tag{24a}$$
$$\mathbf{z}_1^k = \mathbf{z}_4^k + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y}_4^k - \mathbf{b}). \tag{24b}$$

2. *Assume that* $\mathbf{prox}_{\lambda F(\cdot)}$ *is affine. Given the sequences* $\mathbf{x}_1^k$, $\mathbf{z}_1^k$, *and* $\mathbf{y}_1^k$ *of Algorithm 1, if* $\mathbf{x}_1^0$ *and* $\mathbf{z}_1^0$ *satisfy* $-\mathbf{z}_1^0 \in \partial F(\mathbf{A}\mathbf{x}_1^0)$, *then we can initialize Algorithm 5 with* $\mathbf{y}_4^0 = \mathbf{y}_1^1$ *and* $\mathbf{z}_4^0 = \mathbf{z}_1^0 + \lambda^{-1}(\mathbf{A}\mathbf{x}_1^0 + \mathbf{B}\mathbf{y}_1^1 - \mathbf{b})$, *and recover the sequences* $\mathbf{y}_4^k$ *and* $\mathbf{z}_4^k$ *of Algorithm 5 through*

$$\mathbf{y}_4^k = \mathbf{y}_1^{k+1}, \tag{25a}$$
$$\mathbf{z}_4^k = \mathbf{z}_1^k + \lambda^{-1}(\mathbf{A}\mathbf{x}_1^k + \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b}). \tag{25b}$$

Proof. We prove Part 1 only by induction. (The proof for the other part is similar.) The initialization of Algorithm 1 clearly follows (24) at $k = 0$. Suppose that (24) holds at $k \geq 0$. We shall show that (24) holds at $k + 1$. We first show from the affine property of $\mathbf{prox}_{\lambda G(\cdot)}$:

$$\mathbf{B}\mathbf{y}_1^{k+1} = 2\mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{B}\mathbf{y}_4^k. \tag{26}$$

The optimization subproblems for $\mathbf{y}_1$ and $\mathbf{y}_4$ in Algorithms 1 and 5, respectively, are as follows:

$$\mathbf{y}_1^{k+1} = \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x}_1^k + \mathbf{B}\mathbf{y} - \mathbf{b} + \lambda\mathbf{z}_1^k\|_2^2,$$
$$\mathbf{y}_4^{k+1} = \arg\min_{\mathbf{y}} g(\mathbf{y}) + (2\lambda)^{-1}\|\mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y} - \mathbf{b} + \lambda\mathbf{z}_4^k\|_2^2.$$

Following the definition of G in (1), we have

$$\mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b} = \mathbf{prox}_{\lambda G(\cdot)}(-\mathbf{A}\mathbf{x}_1^k - \lambda\mathbf{z}_1^k), \tag{27a}$$
$$\mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b} = \mathbf{prox}_{\lambda G(\cdot)}(-\mathbf{A}\mathbf{x}_4^{k+1} - \lambda\mathbf{z}_4^k), \tag{27b}$$
$$\mathbf{B}\mathbf{y}_4^k - \mathbf{b} \quad = \mathbf{prox}_{\lambda G(\cdot)}(-\mathbf{A}\mathbf{x}_4^k - \lambda\mathbf{z}_4^{k-1}). \tag{27c}$$

The third step of Algorithm 5 is

$$\mathbf{z}_4^k = \mathbf{z}_4^{k-1} + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^k + \mathbf{B}\mathbf{y}_4^k - \mathbf{b}). \tag{28}$$

(Note that for $k = 0$, the assumption $-\mathbf{z}_4^0 \in \partial G(\mathbf{B}\mathbf{y}_4^0 - \mathbf{b})$ ensures the existence of $\mathbf{z}_4^{-1}$ in (27c) and (28).) Then, (24) and (28) give us

$$\begin{aligned}\mathbf{A}\mathbf{x}_1^k + \lambda\mathbf{z}_1^k &\overset{(24)}{=} \mathbf{A}\mathbf{x}_4^{k+1} + \lambda\mathbf{z}_4^k + \mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y}_4^k - \mathbf{b} \\ &= 2(\mathbf{A}\mathbf{x}_4^{k+1} + \lambda\mathbf{z}_4^k) - (\lambda\mathbf{z}_4^k - \mathbf{B}\mathbf{y}_4^k + \mathbf{b}) \\ &\overset{(28)}{=} 2(\mathbf{A}\mathbf{x}_4^{k+1} + \lambda\mathbf{z}_4^k) - (\mathbf{A}\mathbf{x}_4^k + \lambda\mathbf{z}_4^{k-1}).\end{aligned}$$

Since $\mathbf{prox}_{\lambda G(\cdot)}$ is affine, we have (23). Once we plug in (23): $\mathbf{r}_1 = -\mathbf{A}\mathbf{x}_4^{k+1} - \lambda\mathbf{z}_4^k$, $\mathbf{r}_2 = -\mathbf{A}\mathbf{x}_4^k - \lambda\mathbf{z}_4^{k-1}$, and $2\mathbf{r}_1 - \mathbf{r}_2 = -\mathbf{A}\mathbf{x}_1^k - \lambda\mathbf{z}_1^k$ and then apply (27), we obtain (26).

Next, the third step of Algorithm 5 and (26) give us

$$\begin{aligned}\mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b} + \lambda\mathbf{z}_1^k &\overset{(26)}{=} 2(\mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b}) - (\mathbf{B}\mathbf{y}_4^k - \mathbf{b}) + \lambda\mathbf{z}_4^k + (\mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y}_4^k - \mathbf{b}) \\ &= (\mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b}) + \lambda\mathbf{z}_4^k + (\mathbf{A}\mathbf{x}_4^{k+1} + \mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b}) \\ &= (\mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b}) + \lambda\mathbf{z}_4^{k+1}.\end{aligned}$$

This identity shows that the updates of $\mathbf{x}_1^{k+1}$ and $\mathbf{x}_4^{k+2}$ in Algorithms 1 and 5, respectively, have identical data, and therefore, we recover $\mathbf{x}_1^{k+1} = \mathbf{x}_4^{k+2}$.

Lastly, from the third step of Algorithm 1 and the identities above, it follows that

$$\begin{aligned}\mathbf{z}_1^{k+1} &= \mathbf{z}_1^k + \lambda^{-1}(\mathbf{A}\mathbf{x}_1^{k+1} + \mathbf{B}\mathbf{y}_1^{k+1} - \mathbf{b}) \\ &= \mathbf{z}_1^k + \lambda^{-1}\left(\mathbf{A}\mathbf{x}_4^{k+2} + (\mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b} + \lambda\mathbf{z}_4^{k+1} - \lambda\mathbf{z}_1^k)\right) \\ &= \mathbf{z}_4^{k+1} + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^{k+2} + \mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b}).\end{aligned}$$

Therefore, we obtain (24) at $k+1$. □

Remark 6. *We can avoid the technical condition $-\mathbf{z}_4^0 \in \partial G(\mathbf{B}\mathbf{y}_4^0 - \mathbf{b})$ on Algorithm 5 in Theorem 3 Part 1. When it does not hold, we can use the always-true relation $-\mathbf{z}_4^1 \in \partial G(\mathbf{B}\mathbf{y}_4^1 - \mathbf{b})$ instead; correspondingly, we shall add 1 iteration to the iterates of Algorithm 5, namely, initialize Algorithm 1 with $\mathbf{x}_1^0 = \mathbf{x}_4^2$ and $\mathbf{z}_1^0 = \mathbf{z}_4^1 + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^2 + \mathbf{B}\mathbf{y}_4^1 - \mathbf{b})$ and recover the sequences $\mathbf{x}_1^k$ and $\mathbf{z}_1^k$ of Algorithm 1 through*

$$\mathbf{x}_1^k = \mathbf{x}_4^{k+2}, \tag{29a}$$

$$\mathbf{z}_1^k = \mathbf{z}_4^{k+1} + \lambda^{-1}(\mathbf{A}\mathbf{x}_4^{k+2} + \mathbf{B}\mathbf{y}_4^{k+1} - \mathbf{b}). \tag{29b}$$

Similar arguments apply to the other part of Theorem 3.

5 Application: extended monotropic programming

In this section, we use an example to demonstrate that the equivalent algorithms may still have different per-iteration complexities and the primal-dual algorithm, Algorithm 4, may be preferable over the others. The following extended monotropic program [2] arises in the setting of parallel and distributed computation:

$$\underset{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N}{\text{minimize}} \sum_{i=1}^N f_i(\mathbf{x}_i) \qquad \text{subject to} \quad \sum_{i=1}^N \mathbf{A}_i\mathbf{x}_i = \mathbf{b}, \tag{30}$$

where $\mathbf{x}_i \in \mathbf{R}^{n_i}$, $\mathbf{A}_i \in \mathbf{R}^{m\times n_i}$, and $\mathbf{b} \in \mathbf{R}^m$, for $i = 1, \ldots, N$. To apply ADM, one can convert the problem into the following ADM-ready formulation by introducing variables/constraints $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i$:

$$\begin{cases} \underset{\{\mathbf{x}_i\},\{\mathbf{y}_i\}}{\text{minimize}} & \sum_{i=1}^N f_i(\mathbf{x}_i) + \iota_{\{\mathbf{y}:\sum_{i=1}^N \mathbf{y}_i = \mathbf{b}\}}(\mathbf{y}) \\ \text{subject to} & \mathbf{A}_i\mathbf{x}_i - \mathbf{y}_i = \mathbf{0}. \end{cases} \tag{31}$$

Problem (31) is in the form of (P1) with $\mathbf{x} := (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N)$ and $\mathbf{y} := (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$. Therefore, ADM algorithms can be applied. In particular, Algorithm 1 has the following updates for every i at iteration k (cf. [5]):

$$\mathbf{y}_i^{k+1} = \frac{\mathbf{b}}{N} + \mathbf{A}_i\mathbf{x}_i^k + \lambda\mathbf{z}_i^k - \frac{1}{N}\left\{\sum_{j=1}^N \mathbf{A}_j\mathbf{x}_j^k + \lambda\mathbf{z}_j^k\right\}, \tag{32a}$$

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + (2\lambda)^{-1}\left\|\mathbf{A}_i\mathbf{x}_i - \mathbf{y}_i^{k+1} + \lambda\mathbf{z}_i^k\right\|_2^2, \tag{32b}$$

$$\mathbf{z}_i^{k+1} = \mathbf{z}_i^k + \lambda^{-1}(\mathbf{A}_i\mathbf{x}_i^{k+1} - \mathbf{y}_i^{k+1}). \tag{32c}$$

Once $\sum_{j=1}^N \mathbf{A}_j\mathbf{x}_j^k + \lambda\mathbf{z}_j^k$ is computed, the above three steps rely on data of subscript i only, so they can be carried out for $i = 1, \ldots, N$ *in parallel*.

Algorithm 4 for problem (31) has the following updates for every i at iteration k:

$$\bar{\mathbf{u}}_i^k = 2\mathbf{u}_i^k - \mathbf{u}_i^{k-1}, \tag{33a}$$

$$\mathbf{y}_i^{k+1} = \frac{\mathbf{b}}{N} + \mathbf{y}_i^k + \lambda\bar{\mathbf{u}}_i^k - \frac{1}{N}\left\{\sum_{j=1}^N \mathbf{y}_j^k + \lambda\bar{\mathbf{u}}_j^k\right\}, \tag{33b}$$

$$\mathbf{u}_i^{k+1} = \arg\min_{\mathbf{u}_i} f_i^*(-\mathbf{A}_i^*\mathbf{u}_i) + (\lambda/2)\left\|\mathbf{u}_i - \mathbf{u}_i^k + \lambda^{-1}\mathbf{y}_i^{k+1}\right\|_2^2. \tag{33c}$$

Likewise, the above three steps can be carried out for $i = 1, \ldots, N$ *in parallel* except that computing $\sum_{j=1}^N \mathbf{A}_j\mathbf{x}_j^k + \lambda\mathbf{z}_j^k$ requires data of subscripts $j = 1, \ldots, N$. In the distributed setting, the summation term requires communication.

The following corollary, which is a direct result of Theorem 2, establishes the equivalence between iterations (32) and (33).

Corollary 3. *Suppose that* $\mathbf{A}_i\mathbf{x}_i^0 = \lambda(\mathbf{u}_i^0 - \mathbf{u}_i^{-1}) + \mathbf{y}_i^0$ *and* $\mathbf{z}_i^0 = \mathbf{u}_i^0$ *for* $i = 1, 2, \cdots, N$. *Then, iterations* (32) *and* (33) *are equivalent with the following identities for all* $k \geq 0$*:*

$$\mathbf{A}_i\mathbf{x}_i^k = \lambda(\mathbf{u}_i^k - \mathbf{u}_i^{k-1}) + \mathbf{y}_i^k, \qquad \mathbf{z}_i^k = \mathbf{u}_i^k.$$

When (33c) is easy to solve, the complexity of algorithm (33) is smaller than that of algorithm (32).

5.1 An example with different complexities

Problem. In problem (30), let $f_i(\cdot) = (1/2)\|\cdot\|_2^2$ and $\mathbf{A}_i\mathbf{A}_i^* = \mathbf{I}$ for $i = 1, 2, \cdots, N$. The problem becomes finding the solution $\mathbf{x}$ satisfying the constraint $\sum_{i=1}^N \mathbf{A}_i\mathbf{x}_i = \mathbf{b}$ with the minimal ℓ_2 norm.
Primal-dual iteration: Since subproblem (33c) can be solved analytically, iteration (33) simplifies to:

$$\bar{\mathbf{u}}_i^k = 2\mathbf{u}_i^k - \mathbf{u}_i^{k-1}, \tag{34a}$$

$$\mathbf{y}_i^{k+1} = \frac{\mathbf{b}}{N} + \mathbf{y}_i^k + \lambda\bar{\mathbf{u}}_i^k - \frac{1}{N}\left\{\sum_{j=1}^N \mathbf{y}_j^k + \lambda\bar{\mathbf{u}}_j^k\right\}, \tag{34b}$$

$$\mathbf{u}_i^{k+1} = (\lambda\mathbf{u}_i^k - \mathbf{y}_i^{k+1})/(\lambda + 1). \tag{34c}$$

The complexity of (34) for each i and k is $10m$ flops, plus the communication cost if the summation is taken in a distributed setting. To see this: (34a) has $2m$ flops; (34b) has $2m + 3m$ flops, ignoring the summation over j; (34c) has $3m$ flops. In addition, upon termination, we need an additional step to obtain $\mathbf{x}_i$ by solving

$$\underset{\mathbf{x}_i}{\text{minimize}}\, \|\mathbf{x}_i\|_2^2 \quad \text{subject to} \quad \mathbf{A}_i\mathbf{x}_i = \mathbf{y}_i,$$

which can be done analytically by $\mathbf{x}_i = \mathbf{A}_i^*\mathbf{y}_i$ for mn_i flops for each i.
ADM iteration: Since subproblem (32b) can be solved analytically, iteration (32) reduces to:

$$\mathbf{y}_i^{k+1} = \frac{\mathbf{b}}{N} + \mathbf{A}_i\mathbf{x}_i^k + \lambda\mathbf{z}_i^k - \frac{1}{N}\left\{\sum_{j=1}^N \mathbf{A}_j\mathbf{x}_j^k + \lambda\mathbf{z}_j^k\right\}, \tag{35a}$$

$$\mathbf{x}_i^{k+1} = (\lambda\mathbf{I} + \mathbf{A}_i^*\mathbf{A}_i)^{-1}\mathbf{A}_i^*(\mathbf{y}_i^{k+1} - \lambda\mathbf{z}_i^k), \tag{35b}$$

$$\mathbf{z}_i^{k+1} = \mathbf{z}_i^k + \lambda^{-1}(\mathbf{A}_i\mathbf{x}_i^{k+1} - \mathbf{y}_i^{k+1}). \tag{35c}$$

We can store $(\lambda\mathbf{I} + \mathbf{A}_i^*\mathbf{A}_i)^{-1}\mathbf{A}_i^*$ for each i. Note that in the distributed setting, the summation in (35a) has the same cost as that in (34b). Excluding the summation, the complexity of (35) for each i and k is $2mn_i + 10m$

flops, which is calculated as follows: (35a) has $mn_i + 2m + 3m$ flops; (35b) has $mn_i + 2m$ flops; (35c) has $3m$ flops to find $\mathbf{z}_i$ in (35c). In addition, it needs a preprocessing step to obtain $(\lambda\mathbf{I}+\mathbf{A}_i^*\mathbf{A}_i)^{-1}\mathbf{A}_i^* = \mathbf{A}_i^*(\lambda\mathbf{I}+\mathbf{I})^{-1}$ for mn_i flops for each i.

To summarize, we can compare (34) and (35) as follows:

- For each i and k, (34) has $10m$ flops comparing to the $2mn_i + 10m$ flops of (35). Since (34) does not explicitly update $\mathbf{x}_i$, it saves $2mn_i$ flops. Such saving is large and important when n_i's are large.
- In addition, (34) has a post-step and (35) has a pre-step, both of which have mn_i flops for each i.
- In the distributed setting, while the communication cost is the same for both algorithms, (34) is still preferred over (35) since the $10m$ flops of (34) does not depend on i and thus is good for load balancing.

6 Application: Total variation image denoising

ADM (or Split Bregman [15]) has been applied to many image processing applications, and we apply the previous equivalence results of ADM to derive several equivalent algorithms for the total variation image denoising.

The total variation (ROF model [17]) applied on image denoising is

$$\underset{x\in BV(\Omega)}{\text{minimize}} \int_\Omega |Dx| + \frac{\alpha}{2}\|x-b\|_2^2$$

where x stands for an image and $BV(\Omega)$ is the set of all bounded variation functions on Ω. The first term is known as the total variation of x, minimizing which tends to yield a piece-wise constant solution. The discrete version is as follows:

$$\underset{\mathbf{x}}{\text{minimize}}\, \|\nabla\mathbf{x}\|_{2,1} + \frac{\alpha}{2}\|\mathbf{x}-\mathbf{b}\|_2^2.$$

Without loss of generality, we consider the two-dimensional image $\mathbf{x}$, and the discrete total variation $\|\nabla\mathbf{x}\|_{2,1}$ of image $\mathbf{x}$ is defined as

$$\|\nabla\mathbf{x}\|_{2,1} = \sum_{ij} |(\nabla\mathbf{x})_{ij}|,$$

where $|\cdot|$ is the 2-norm of vector $(\nabla\mathbf{x})_{ij}$. The equivalent ADM-ready form [15, Equation (3.1)] is

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}}\ \ \|\mathbf{y}\|_{2,1} + \frac{\alpha}{2}\|\mathbf{x}-\mathbf{b}\|_2^2 \qquad \text{subject to } \mathbf{y}-\nabla\mathbf{x} = \mathbf{0}, \tag{36}$$

and its problem in the ADM-ready form [3, Equation (8)] is

$$\underset{\mathbf{v},\mathbf{u}}{\text{minimize}}\ \ \frac{1}{2\alpha}\|\text{div }\mathbf{u} + \alpha\mathbf{b}\|_2^2 + \iota_{\{\mathbf{v}:\|\mathbf{v}\|_{2,\infty}\le 1\}}(\mathbf{v}) \qquad \text{subject to } \mathbf{u}-\mathbf{v} = \mathbf{0}, \tag{37}$$

where $\|\mathbf{v}\|_{2,\infty} = \max_{ij} |(\mathbf{v})_{ij}|$.

In addition, the equivalent saddle-point problem is

$$\min_{\mathbf{x}}\max_{\mathbf{v}} \quad \frac{1}{2\alpha}\|\mathbf{x}-\mathbf{b}\|_2^2 + \langle\mathbf{v},\nabla\mathbf{x}\rangle - \iota_{\{\mathbf{v}:\|\mathbf{v}\|_{2,\infty}\le 1\}}(\mathbf{v}). \tag{38}$$

We list the following equivalent algorithms for solving the total variation image denoising problem. The equivalence result stated in Corollary 4 can be obtained from theorems 1-3.

1. Algorithm 1 (primal ADM) on (36) is

$$\mathbf{x}_1^{k+1} = \arg\min_{\mathbf{x}} \frac{\alpha}{2}\|\mathbf{x}-\mathbf{b}\|_2^2 + (2\lambda)^{-1}\|\nabla\mathbf{x} - \mathbf{y}_1^k + \lambda\mathbf{z}_1^k\|_2^2, \tag{39a}$$

$$\mathbf{y}_1^{k+1} = \arg\min_{\mathbf{y}} \|\mathbf{y}\|_{2,1} + (2\lambda)^{-1}\|\nabla\mathbf{x}_1^{k+1} - \mathbf{y} + \lambda\mathbf{z}_1^k\|_2^2, \tag{39b}$$

$$\mathbf{z}_1^{k+1} = \mathbf{z}_1^k + \lambda^{-1}(\nabla\mathbf{x}_1^{k+1} - \mathbf{y}_1^{k+1}). \tag{39c}$$

2. Algorithm 3 (dual ADM) on (37) is

$$\mathbf{u}_2^{k+1} = \arg\min_{\mathbf{u}} \frac{1}{2\alpha}\|\text{div } \mathbf{u} + \alpha\mathbf{b}\|_2^2 + \frac{\lambda}{2}\|\mathbf{v}_2^k - \mathbf{u} + \lambda^{-1}\mathbf{z}_2^k\|_2^2, \tag{40a}$$

$$\mathbf{v}_2^{k+1} = \arg\min_{\mathbf{v}} \iota_{\{\mathbf{v}:\|\mathbf{v}\|_{2,\infty}\le 1\}}(\mathbf{v}) + \frac{\lambda}{2}\|\mathbf{v} - \mathbf{u}_2^{k+1} + \lambda^{-1}\mathbf{z}_2^k\|_2^2, \tag{40b}$$

$$\mathbf{z}_2^{k+1} = \mathbf{z}_2^k + \lambda(\mathbf{v}_2^{k+1} - \mathbf{u}_2^{k+1}). \tag{40c}$$

3. Algorithm 4 (primal-dual) on (38) is

$$\bar{\mathbf{v}}_3^k = 2\mathbf{v}_3^k - \mathbf{v}_3^{k-1} \tag{41a}$$

$$\mathbf{x}_3^{k+1} = \arg\min_{\mathbf{x}} \frac{\alpha}{2}\|\mathbf{x}-\mathbf{b}\|_2^2 + (2\lambda)^{-1}\|\nabla\mathbf{x} - \nabla\mathbf{x}_3^k + \lambda\bar{\mathbf{v}}_3^k\|_2^2, \tag{41b}$$

$$\mathbf{v}_3^{k+1} = \arg\min_{\mathbf{v}} \iota_{\{\mathbf{v}:\|\mathbf{v}\|_{2,\infty}\le 1\}}(\mathbf{v}) - \langle\mathbf{v}, \nabla\mathbf{x}_3^{k+1}\rangle + \frac{\lambda}{2}\|\mathbf{v} - \mathbf{v}^k\|_2^2. \tag{41c}$$

4. Algorithm 5 (primal ADM with order swapped) on (36) is

$$\mathbf{y}_4^{k+1} = \arg\min_{\mathbf{y}} \|\mathbf{y}\|_{2,1} + (2\lambda)^{-1}\|\nabla\mathbf{x}_4^k - \mathbf{y} + \lambda\mathbf{z}_4^k\|_2^2, \tag{42a}$$

$$\mathbf{x}_4^{k+1} = \arg\min_{\mathbf{x}} \frac{\alpha}{2}\|\mathbf{x}-\mathbf{b}\|_2^2 + (2\lambda)^{-1}\|\nabla\mathbf{x} - \mathbf{y}_4^{k+1} + \lambda\mathbf{z}_4^k\|_2^2, \tag{42b}$$

$$\mathbf{z}_4^{k+1} = \mathbf{z}_4^k + \lambda^{-1}(\nabla\mathbf{x}_4^{k+1} - \mathbf{y}_4^{k+1}). \tag{42c}$$

Corollary 4. *Let* $\mathbf{x}_4^0 = \mathbf{b} + \alpha^{-1}\text{div } \mathbf{z}_4^0$. *If the initialization for all algorithms* (39)-(42) *satisfy* $\mathbf{y}_1^0 = -\mathbf{z}_2^0 = \nabla\mathbf{x}_3^0 - \lambda(\mathbf{v}_3^0 - \mathbf{v}_3^{-1}) = \mathbf{y}_4^1$ *and* $\mathbf{z}_1^0 = \mathbf{v}_2^0 = \mathbf{v}_3^0 = \mathbf{z}_4^0 + \lambda^{-1}(\nabla\mathbf{x}_4^0 - \mathbf{y}_4^1)$. *Then for* $k \ge 1$, *we have the following equivalence between the iterations of the four algorithms:*

$$\begin{array}{llll} \mathbf{y}_1^k & = -\mathbf{z}_2^k & = \nabla\mathbf{x}_3^k - \lambda(\mathbf{v}_3^k - \mathbf{v}_3^{k-1}) & = \mathbf{y}_4^{k+1}, \\ \mathbf{z}_1^k & = \mathbf{v}_2^k & = \mathbf{v}_3^k & = \mathbf{z}_4^k + \lambda^{-1}(\nabla\mathbf{x}_4^k - \mathbf{y}_4^{k+1}). \end{array}$$

Remark 7. *In any of the four algorithms, the* ∇ *or* div *operator is separated in a different subproblem from the term* $\|\cdot\|_{2,1}$ *or its dual norm* $\|\cdot\|_{2,\infty}$. *The* ∇ *or* div *operator is translation invariant so their subproblems can be solved by a diagonalization trick [18]. The subproblems involving the term* $\|\cdot\|_{2,1}$ *or the indicator function* $\iota_{\{\mathbf{v}:\|\mathbf{v}\|_{2,\infty}\le 1\}}$ *have closed-form solutions. Therefore, in addition to the equivalence results, all the four algorithms have essentially the same per-iteration costs.*

Acknowledgements

This work is supported by NSF Grants DMS-1349855 and DMS-1317602 and ARO MURI Grant W911NF-09-1-0383. We thank Jonathan Eckstein for bringing his early work [8, Chapter 3.5] and [9] to our attention.

References

[1] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, Springer, 2011.

[2] D. Bertsekas, *Extended monotropic programming and duality*, Journal of Optimization Theory and Applications, 139 (2008), pp. 209–225.

[3] A. Chambolle, *An algorithm for total variation minimization and applications*, J. Math. Imaging Vis., 20 (2004), pp. 89–97.

[4] A. Chambolle and T. Pock, *A first-order primal-dual algorithm for convex problems with applications to imaging*, Journal of Mathematical Imaging and Vision, 40 (2011), pp. 120–145.

[5] W. Deng, M.-J. Lai, Z. Peng, and W. Yin, *Parallel multi-block admm with o(1/k) convergence*, ArXiv e-prints 1312.3040, (2013).

[6] W. Deng and W. Yin, *On the global and linear convergence of the generalized alternating direction method of multipliers*, Rice University CAAM Technical Report, (2012).

[7] J. Douglas, Jim and J. Rachford, H. H., *On the numerical solution of heat conduction problems in two and three space variables*, Transactions of the American Mathematical Society, 82 (1956), pp. pp. 421–439.

[8] J. Eckstein, *Splitting methods for monotone operators with applications to parallel optimization*, PhD thesis, Massachusetts Institute of Technology, 1989.

[9] J. Eckstein and M. Fukushima, *Some reformulations and applications of the alternating direction method of multipliers*, in Large scale optimization, Springer US, 1994, pp. 115–134.

[10] E. Esser, X. Zhang, and T. Chan, *A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science*, SIAM Journal on Imaging Sciences, 3 (2010), pp. 1015–1046.

[11] M. Fukushima, *The primal douglas-rachford splitting algorithm for a class of monotone mappings with application to the traffic equilibrium problem*, Mathematical Programming, 72 (1996), pp. 1–15.

[12] D. Gabay, *Applications of the method of multipliers to variational inequalities*, in Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems, M. Fortin and R. Glowinski, eds., North-Holland: Amsterdam, Amsterdam, 1983.

[13] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Comput. Math. Appl., 2 (1976), pp. 17–40.

[14] R. Glowinski and A. Marroco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires*, Rev. Française d'Automat. Inf. Recherche Opérationelle, 9 (1975), pp. 41–76.

[15] T. Goldstein and S. Osher, *The split bregman method for l1-regularized problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 323–343.

[16] P. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM Journal on Numerical Analysis, 16 (1979), pp. 964–979.

[17] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Physica D: Nonlinear Phenomena, 60 (1992), pp. 259 – 268.

[18] Y. Wang, J. Yang, W. Yin, and Y. Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 248–272.

[19] Y. Xiao, H. Zhu, and S.-Y. Wu, *Primal and dual alternating direction algorithms for l1-l1-norm minimization problems in compressive sensing*, Comput. Optim. Appl., 54 (2013), pp. 441–459.

[20] J. Yang and Y. Zhang, *Alternating direction algorithms for ℓ_1-problems in compressive sensing*, SIAM journal on scientific computing, 33 (2011), pp. 250–278.